Remarks

Status of application

Claims 1-47 were examined and stand rejected in view of prior art. The claims have been amended to further clarify Applicant's invention. Reexamination and reconsideration are respectfully requested.

The invention

A security system with methodology for interprocess communication control is described. In one embodiment, a method for controlling interprocess communication is provided that includes steps of: defining rules indicating which system services a given application can invoke; trapping an attempt by a particular application to invoke a particular system service; identifying the particular application that is attempting to invoke the particular system service; and based on identity of the particular application and on the rules indicating which system services a given application can invoke, blocking the attempt when the rules indicate that the particular application cannot invoke the particular system service.

General

Minor non-art amendments have been made to the specification (Claims heading) and to claim 13, for purposes of improving readability.

Prior art rejections

A. Section 102: Andrews

Claims 1-47 stand rejected under 35 U.S.C. 102(e) as being anticipated by Andrews (US 6,574,736). The Examiner's rejection of claim 1 is representative:

> Andrews discloses a method for controlling interprocess communication, the method comprising:
>
> defining rules indicating which system services a given application can invoke; (col. 9, lines 34-38 and 49-55) trapping an attempt by a particular application to invoke a particular system service; (col. 15, lines

20-22 and col .21, lines 35-37)

identifying the particular application that is attempting to invoke the particular system service; and (col. 20, lines 15-16 and col. 22, lines 13-18 and 30-32)

based on identity of the particular application and on the rules indicating which system services a given application can invoke (col. 14, lines 56-59), blocking the attempt when the rules indicate that the particular application cannot invoke the particular system service. (col. 22, lines 1-9) .

Andrews' teaching is directed to "composable roles" -- that is, defining access privileges for users. As described by Andrews' Abstract, in Andrews's system an application developer grants access privileges to application processing services in an object-based application by defining logical classes of users called "roles." When the application is deployed on a host computer system, an administrator populates the roles with users and groups (of users) recognized by the host computer system. At runtime, a user is not permitted access to a processing service unless the user is a member of a permitted role for the processing service. To ease administration, two or more roles can be composed. In one implementation, roles are associated with a separate composite role. The administrator can then populate the composite role instead of individually populating each of the roles associated with the composite role. The foregoing approach of Andrews has little in common with Applicant's invention.

Andrews describes in detail the problem that he is addressing and which developers face: finding a way to implement a security scheme efficiently in object-oriented applications. For example, a developer of a banking application may want to prevent tellers (i.e., specific group of users) from accessing an object for changing a customer's deposit balance. However, the developer may be designing an application for use by a wide variety of banking organizations with different organizational structures and different employees (again, note that this is user-based control). It would be a burdensome task for the application developer to customize the banking application for each of the banking organizations. Andrews emphasizes that if user identities specific to

the banking organization were placed in the objects' logic, maintaining the application would require the further burden of changing the logic to account for employee turnover or reorganization. (See, e.g., Andrews at column 2, lines 31-44). His particular focus is on controlling **user access** in the context of object-oriented applications, where functionality may be divvied up among different components, such as an "extended application" having separate shipping and billing components. In the extended application, the developer of the shipping component may have defined a user role of "manager" while the developer of the billing component may have defined a role of "mgr". The particular solution provided by Andrews to this problem is to apply his "composing roles." An administrator can associate a first role with a second role, which takes on the population of the first role. Subsequent changes to the first role's population are automatically implemented to the second role's population. Clearly, Andrews' teaching is directed to controlling user access.

Applicant's invention provides a security system with methodology for controlling interprocess communication (IPC) -- communications that may occur between processes that operate on computer systems, including ones that may operate autonomously (i.e., without user initiation or control). IPC is the means by which one process may access another process for the purpose of sending it a message. IPC occurs directly between processes (e.g., between two applications, or between an application and a service) without any involvement of the user. In fact, the user typically will not even be aware that IPC is occurring between two processes. Importantly, the specific vulnerability addressed by Applicant's invention does not even involve the user, but instead involves a malicious process (malware) that has been placed on the user's machine. In this scenario, the user's own privileges (access rights) are not at issue: the user is assumed to be a bona fide user with appropriate authority to use his or her computer. Instead, the issue addressed by Applicant's invention is that the user has unfortunately picked up malware, such as by accidentally downloading it from the Internet (e.g., by well known conduits, such as Trojan horse programs, viruses, worms, and the like). Andrews, in contrast, is concerned about controlling users for preventing an unauthorized user (i.e., a human) from gaining access to information or resources that he or she is not authorized for. Although Applicant's invention and Andrews' teaching pertain generally to the field of

computer security, a thorough review reveals that each is directed to an aspect of computer security that is completely different from the other.

To be sure, Applicant's invention shares some features with Andrews' system as both are in the field of computer security; both share general features with other computer security products as well. For example, both may employ security rules for defining conditions. Also, both may block a given access attempt, if an applicable rule or condition is not met. However, apart from general security features (present in all security-related solutions), Applicant's claimed invention includes specific features pertaining to controlling interprocess communication that may occur between two separate processes completely irrespective of the user or his or her access privileges. These features are not taught or suggested by Andrews. In order to highlight this distinction, Applicant's independent claims 1, 14, and 25 have been amended to more explicitly recite the controlling of interprocess communication between processes. (Independent claim 36 already explicitly recited the controlling of interprocess communication.)

Turning now to the specific teachings of Andrews cited by the Examiner, one finds that the specific points of Andrews highlighted by the Examiner bears little resemblance to Applicant's claim limitations. For example, for Applicant's claim limitation of "defining rules indicating which system services a given application can invoke," the Examiner points to Andrews at column 9, lines 34-38 and 49-55. There, one finds that Andrews discusses the creation of a "composite role," where "a composite role is a role bound to at least one role," and "a role is a logical class of users having access privileges to processing services of an application." Andrews also describes that a "composite role" may include an "application role," but note that an application role is simply a user role bound to a particular application. For example, Andrews states the example: "Since the 'tellers' role and the 'managers' role are associated with a particular application and are defined as development of the application they are sometimes called 'application roles.'" (Andrews at column 9, lines 62-65.) Here, Andrews is concerned about particular functionality of an application, such as "an object method for changing a customer's address" (see Andrews at column 9, lines 56-57). Andrews does not discuss interprocess communication between different processes (e.g., between two applications,

or between an application and a system service), nor would one expect to find such a discussion since Andrews is focused on controlling access of users. Andrews is concerned about an unauthorized or malicious human user gaining inappropriate access to sensitive computing services. Andrews is completely unconcerned about the scenario of an authorized user whose computer has been infected with malware that is attempting unauthorized interprocess communication.

In order to further highlight this distinction of Applicant's invention, Applicant's claim 1 has now been amended to recite (shown in amended form):

> defining rules indicating which system services a given application can invoke using interprocess communication to invoke said system services;

Here, the concern being addressed is that a given application -- running on an authorized user's computer -- may potentially use interprocess communication to invoke system services in an inappropriate manner. These are "bad acts" that may occur wholly between computer processes irrespective of the user -- that is, irrespective of any user status, role, or even user awareness of the situation. The user himself or herself is assumed to be a fully authorized user and thus is not the threat being addressed (unlike Andrews, who is essentially describing a mechanism for keeping out bad (unauthorized) human users from accessing particular services of an application).

Regarding Applicant's claim limitation of "trapping an attempt by a particular application to invoke a particular system service," the Examiner points to Andrews at column 15, lines 20-22 and column 21, lines 35-37. In the field of computer science, many scenarios arise where something needs to be trapped or intercepted, and Applicant certainly makes no claim to have invented the notion by itself. In the above section cited by the Examiner, however, note particularly that this intercepting or trapping described by Andrews is specific user-based intercepting. Andrews explains: "To indicate which user initiated execution of the objects, an identity is associated with calls from the client object 706 (e.g., the identity of a logged on user or an identity indicating the system user) as described in more detail below." (Andrews at column 15, lines 28-32). Clearly,

Andrews is describing a scenario of a user attempting to initiate execution of objects. Andrews is not describing a scenario of an autonomous process (i.e., something not under user control, such as malware) attempting to initiate or invoke another process's services.

Further, regarding Applicant's limitation of "blocking the attempt when the rules indicate **that the particular application cannot invoke** the particular system service," the Examiner points to Andrews at column 14, lines 56-59. There, however, one finds that Andrews describes: "At run time, a security service monitors calls to objects and limits access to those **user** identities that are members of the role associated with the method, interface, object, or application being called." (Emphasis added.) Significantly, Andrews has no notion that a particular application (e.g., malware) itself may be unauthorized, and therefore may need its access to sensitive system services controlled. (Not surprisingly, a text search of Andrews' patent reveals absolutely no discussion whatsoever of "malware", "viruses", "Trojan horses", "worms", or the like, nor does Andrews specification even mention "interprocess communication".) Andrews instead focuses his teachings on controlling user access, and as a result fails to provide sufficient teaching or suggestion to anticipate Applicant's claimed invention.

All told, Applicant's claims set forth a patentable advance in the area of controlling access of potentially "bad" applications or processes. Applicant's claims have been amended to highlight the specific features of Applicant's invention that address the vulnerability posed by interprocess communication, that one application or process may attempt to use interprocess communication to thwart security measures. In view of the foregoing remarks (and in light of clarifying amendments made to the claims), it is respectfully submitted that the claims distinguish over Andrews and any rejection under Section 102 is overcome.

Any dependent claims not explicitly discussed are believed to be allowable by virtue of dependency from Applicant's independent claims, as discussed in detail above.

Conclusion

In view of the foregoing remarks and the amendment to the claims, it is believed

that all claims are now in condition for allowance. Hence, it is respectfully requested that the application be passed to issue at an early date.

If for any reason the Examiner feels that a telephone conference would in any way expedite prosecution of the subject application, the Examiner is invited to telephone the undersigned at 408 884 1507.

Respectfully submitted,

Date: January 16, 2007

John A. Smart; Reg. No. 34,929
Attorney of Record

408 884 1507
815 572 8299 FAX